58

UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER OF PATENTS AND TRADEMARKS
Washington, D.C. 20231
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/496,844	02/02/2000	Patrick Knebel	10971393-1	6757

22879 7590 08/27/2002

HEWLETT PACKARD COMPANY
P O BOX 272400, 3404 E. HARMONY ROAD
INTELLECTUAL PROPERTY ADMINISTRATION
FORT COLLINS, CO 80527-2400

EXAMINER	
HUISMAN, DAVID J	
ART UNIT	PAPER NUMBER
2183	

DATE MAILED: 08/27/2002

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	**Application No.** 09/496,844	**Applicant(s)** KNEBEL ET AL.	
	Examiner David J. Huisman	**Art Unit** 2183	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133).
- Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 02 February 2000 .

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☐ Claim(s) _____ is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-18 is/are rejected.

7)☒ Claim(s) 13,14 and 16 is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☒ The specification is objected to by the Examiner.

10)☒ The drawing(s) filed on 02 February 2000 is/are: a)☐ accepted or b)☒ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

11)☐ The proposed drawing correction filed on _____ is: a)☐ approved b)☐ disapproved by the Examiner.

If approved, corrected drawings are required in reply to this Office action.

12)☐ The oath or declaration is objected to by the Examiner.

Priority under 35 U.S.C. §§ 119 and 120

13)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____ .

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

14)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. § 119(e) (to a provisional application).

a) ☐ The translation of the foreign language provisional application has been received.

15)☐ Acknowledgment is made of a claim for domestic priority under 35 U.S.C. §§ 120 and/or 121.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☒ Information Disclosure Statement(s) (PTO-1449) Paper No(s) 2 .

4) ☐ Interview Summary (PTO-413) Paper No(s). _____ .

5) ☐ Notice of Informal Patent Application (PTO-152)

6) ☐ Other: .

DETAILED ACTION

1. Claims 1-18 have been examined.

Papers Submitted

2. It is hereby acknowledged that the following papers have been received and placed of record in the file: IDS as received on 10/26/00.

Specification

3. The title of the invention is not descriptive. A new title is required that is clearly indicative of the invention to which the claims are directed.

4. The disclosure is objected to because of the following informalities: On page 9, lines 1 and 2, the applicant specifies "op8" as being an operation that must be flushed. However, on page 8, "op8" is not mentioned. Instead, the applicant specifies operations up to and including "op6". Therefore, please replace both instances of "op8" on page 9, lines 1 and 2 with --op6--.

Appropriate correction is required.

Drawings

5. The drawings are objected to under 37 CFR 1.83(a) because Figure 3 fails to show all labeled components as described in the specification. Please label the boxes representing the MXCSR register (130) and the two functional units (170, 172). A proposed drawing correction or corrected drawings are required in reply to the Office action to avoid abandonment of the application. The objection to the drawings will not be held in abeyance.

Claim Objections

6. Claim 13 is objected to because it claims the method of claim 12. However, claim 12 claims a computer system, not a method. Therefore, please replace the word "method" in claim 13 on page 11, line 31, with --computer system--.

7. Claim 14 is objected to because it claims the method of claim 13. However, as discussed in the objection of claim 13, all dependent claims of claim 12 should claim a computer system as opposed to the method. Therefore, please replace the word "method" in claim 14 on page 12, line 1, with --computer system--.

8. Claim 16 is objected to because of the following informalities: The word "emulated" on page 12, line 8, should be replaced with --emulate--. Appropriate correction is required.

Claim Rejections - 35 USC § 103

9. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

10. Claims 1-8 and 10-17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Abdallah et al., U.S. Patent No. 6,233,671 B1 (herein referred to as Abdallah1) in view of Abdallah et al., U.S. Patent No. 6,085,312 (herein referred to as Abdallah2).

11. Referring to claim 1, Abdallah1 has taught a method for processing software instructions comprising:

a) decomposing a macroinstruction into a plurality of microinstructions. See the title.

b) issuing at least two of the plurality of microinstructions in parallel. See FIG.3 and FIG.4B.

c) Abdallah1 has not explicitly taught determining whether an exception occurs in any of the at least two of a plurality of microinstructions. However, Abdallah2 has taught the concept of checking for an exception in each of the microinstructions. See column 6, lines 52-59. It is well known in the art that processors check for exceptions while executing instructions. If exceptions were not monitored, then illegal operations would be performed (such as divide-by-zero) or incorrect results would be obtained (due to overflow, for instance). Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to check each microinstruction in Abdallah1's system for an exception in order to assure the processor handles each instruction properly.

d) Abdallah1 has not explicitly taught canceling the at least two of a plurality of microinstructions if an exception occurs in any of the at least two of a plurality of microinstructions. On the other hand, Abdallah2 has taught the concept of flushing the system if exceptions occur while executing microinstructions. See column 10, lines 21-24. A person of ordinary skill in the art would have realized that if the system were not flushed when an exception occurs, then an undesired result would be written into a register. This would result in having to perform an "undo" operation in order to return to the state before the incorrect result was written. Abdallah2 has focused on flushing the system when a second microinstruction triggers an exception. However, a person of ordinary skill in the art would have also recognized that if an exception is triggered by the first microinstruction, the system should be flushed because it would be pointless to execute the second instruction if there is already a problem with the first instruction. Therefore, in order to save processing time, it would have been obvious to one of ordinary skill in the art at the time of the invention to cancel the at least two of a plurality of microinstructions in Abdallah1's system if an exception occurs in any of the at least two of a plurality of microinstructions.

12. Referring to claim 2, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 1. Abdallah1 has further taught the execution of at least two of a plurality of microinstructions. See FIG.3 and FIG.4B.

13. Referring to claim 3, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 2. Abdallah1 has taught that at least two of a plurality of microinstructions are executed on separate execution units, but appear as though they were executed on a single execution unit. See FIG.3, FIG.4A, and FIG.4B. For example, from FIG.4B, one can see that the instruction ADD X,Y is broken down into 4 microinstructions: ADD X_0 Y_0, ADD X_1 Y_1, ADD X_2 Y_2, and ADD X_3 Y_3. Through the use of two separate execution units, the first two microinstructions are executed in parallel followed by the parallel execution of the second two microinstructions, which can be seen in FIG.4A. Each of the results obtained from the parallel execution is then combined as shown in FIG.3.

14. Referring to claim 4, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 1. In addition, it is well known in the art that if at least two of a plurality of microinstructions are executed in parallel, then they are also executed in the same clock cycle.

15. Referring to claim 5, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 1. Furthermore, Abdallah1 has taught a staggered execution of at least two of a plurality of microinstructions. See FIG.4B and a brief explanation in column 5, lines 22-31.

16. Referring to claim 6, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 1. Abdallah1 has not explicitly taught the implementation of this method in a system emulating SSE instructions. However, a person of ordinary skill in the art would have realized that SIMD is often associated with accelerating floating-point performance in general purpose processors mainly due to the fact that multiple floating-point operations can be executed in parallel. Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to incorporate SSE instructions into Abdallah1's system in order to maximize the efficiency in which the processor executes floating-point operations.

17. Referring to claim 7, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 6. Abdallah1 has further taught a system that allows a single instruction to operate on multiple single-precision floating-point values. In column 5, lines 1-4, Abdallah1 discloses the use of two single-precision ADD execution units and two single-precision MUL execution units. Furthermore, in column 11, lines 11-18, Abdallah1 mentions the use of 128-bit packed floating-point data operands, which are divided into a low half (lower 64 bits) and a high half (upper 64 bits). It is well known in the art that a single-precision floating-point number is a 32-bit number. As shown in FIG.3, since each half is executed in parallel, it can be realized that each adder is operating on a single-precision (32-bit) number. Therefore, it follows that a single instruction in Abdallah1's system operates on single-precision floating-point numbers.

18. Referring to claim 8, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 1. Furthermore, it is well known in the art that processors contain a status register. The status register contains bits that are set or cleared based on the result of an operation. Some of the more common flags are ones that indicate a result of zero, a negative

number, and overflow. These flags can then be checked in conditional situations, such as branches.

19. Referring to claim 10:

a,b) Abdallah1 has taught the parallel execution of two microinstructions with 32-bit operands. See FIG.3 and FIG.4B. Since Abdallah1 has taught 128-bit registers, it follows that X_0, X_1, X_2, X_3, Y_0, Y_1, Y_2, and Y_3 are all 32-bit numbers. Therefore, two separate 32-bit functional units were implemented to perform two separate adds in parallel (X_1+Y_1 and X_0+Y_0). However, a person of ordinary skill in the art would have realized that 64-bit functional units could replace Abdallah1's 32-bit functional units. This would result in being able to provide two microinstructions to emulate a high-half and a low-half SSE operation as opposed to four microinstructions to emulate a high-half and a low-half SSE operation. For instance, if two 64-bit functional units were used, then X_0 and X_1 could be combined to form a 64-bit number A_0 and X_2 and X_3 could be combined to form a 64-bit number A_1. Furthermore, Y_0 and Y_1 could be combined to form a 64-bit number B_0 and Y_2 and Y_3 could be combined to form a 64-bit number B_1. Then, two separate 64-bit instructions (A_1+B_1 and A_0+B_0) could be issued in parallel. Therefore, in order to reduce overall execution time of the microinstructions, it would have been obvious to one of ordinary skill in the art at the time of the invention to use two microinstructions to emulate a high-half and low-half SSE operation and to issue them in parallel.

c) A person of ordinary skill in the art would have recognized that if two operations are to be issued in parallel as described in claim 10(a), then in order to take advantage of instruction parallelism, the high-half and low-half operations would be simultaneously dispatched to a first and second FP unit, respectively. If the instructions were not dispatched simultaneously, they would not be executed in parallel.

d,e) Abdallah1 in view of Abdallah2 has not explicitly taught the generation of a signal from an emulator's hardware to specify that an SSE operation is being emulated. However, a person of ordinary skill in the art would have recognized that a signal is required in order to specify that the microinstructions depend on one another. This signal must be supplied to the first and second FP units because that is where the microinstruction execution is taking place. This signal dictates that when an exception occurs in one functional unit, because the microinstructions are linked, the overall instruction must be dealt with accordingly. Therefore, in order to dictate the linkage and proper handling of microinstructions, it would have been obvious to one of ordinary skill in the art to first generate a signal from the hardware in Abdallah1 in view of Abdallah2's system and send it to the functional units where the execution takes place.

f) Abdallah1 has not explicitly taught determining whether an exception occurs in any of the at least two of a plurality of microinstructions. However, Abdallah2 has taught the concept of checking for an exception in each of the microinstructions. See column 6, lines 52-59. It is well known in the art that processors check for exceptions while executing instructions. If exceptions were not monitored, then illegal operations will be performed (such as divide-by-zero) or incorrect results will be obtained (due to overflow, for instance). Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to check each microinstruction in Abdallah1's system for an exception in order to assure the processor handles each instruction properly.

g) In addition, Abdallah1 has not explicitly taught canceling the at least two of a plurality of microinstructions if an exception occurs in any of the at least two of a plurality of microinstructions. On the other hand, Abdallah2 has taught the concept of flushing the system if exceptions occur while executing microinstructions. See column 10, lines 21-24. A person of ordinary skill in the art would have realized that if the system were not flushed when an exception occurs, then an undesired result could be written into a register. This would result in having to perform an "undo" operation in order to return to the state before the incorrect result was written. Abdallah2 has focused on flushing the system when a second microinstruction triggers an exception. However, when an exception is triggered by the first microinstruction, the system should be flushed because it would be pointless to execute the second instruction if there is already a problem with the first instruction. Therefore, in order to save processing time, it would have been obvious to one of ordinary skill in the art at the time of the invention to cancel the at least two of a plurality of microinstructions in Abdallah1's system if an exception occurs in any of the at least two of a plurality of microinstructions.

h) Abdallah1 in view of Abdallah2 has not explicitly taught the updating of the MXCSR flags based upon the results of the first and second FP units. However, as discussed in the rejection of claim 8 above, a person of ordinary skill in the art would have recognized that processors contain a status register of some type. The MXCSR register contains flags that are common to other processor status registers. These bits (flags) are set and cleared based on results from operations. For instance, if a result of an addition is zero, a flag indicating a zero-result would be set in the status register. Or, perhaps an overflow occurred. A flag in the status register would be set to specify that as well. Conditional statements such as branches then reference these flags in order to determine the direction of the program. Therefore, in order to provide a readable status of the processor so that programs (via branches) flow according to previously obtained results, it would have been obvious to one of ordinary skill in the art at the time of the invention to update flags that are found in the MXCSR register.

20. Referring to claim 11, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 10. Abdallah1 in view of Abdallah2 has not explicitly taught that flushing a result in the other FP unit does not depend on the relative ages of the two microinstructions. However, a person of ordinary skill in the art would have recognized that age is not an issue when flushing the system. Instead, the only concern is whether an exception occurs. If an exception occurs, regardless of the microinstruction ages, the FP units operating on related microinstructions should be flushed to assure illegal operations (such as divide-by-zero) don't execute and also to assure that incorrect results (due to overflow, for instance) are not written to a final result register. Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to flush the FP units if an exception is detected, regardless of the microinstruction ages.

21. Referring to claim 12, it is inherent that a computer system comprises a processor comprising:

a) a floating-point unit. Processors must contain floating-point units in order to perform operations on floating-point numbers.

b) a ROM. Processors contain Read-Only Memory to store essential software of the computer. Because it's non-volatile memory, ROM does not lose its contents when the power is turned off.

Therefore, a ROM chip is used to store control programs for the computer, such as the bootstrap program (which tells the computer how to start and load the operating system).

c) a plurality of floating-point registers. Processors that perform floating-point operations must have floating-point registers in order to hold the floating-point data.

Furthermore, note that lines 23-30 of claim 12 are identical to lines 4-11 of claim 1. Therefore, the rejection of parts [a-d] in claim 12 on lines 23-30, is identical to the rejection of claim 1 above.

22. Referring to claim 13, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 12. It is noted by the examiner that claim 13 is the same as claim 2. Therefore, the rejection of claim 13 is based on the same reasons as mentioned above in the rejection of claim 2.

23. Referring to claim 14, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 13. It is noted by the examiner that claim 14 is the same as claim 3. Therefore, the rejection of claim 14 is based on the same reasons as mentioned above in the rejection of claim 3.

24. Referring to claim 15, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 14. It is noted by the examiner that the only difference between claim 15 and claim 8 is that claim 15 claims that a processor is configured to emulate an instruction set by updating a flag. However, it is inherent that a processor would have to perform the method for processing software instructions that is claimed in claim 8. Therefore, the rejection of claim 15 is based on the same reasons as mentioned above in the rejection of claim 8.

25. Referring to claim 16, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 15. Abdallah1 in view of Abdallah2 has further taught:

a) determining whether an exception occurs in the execution of any of the at least two of a plurality of microinstructions. It is inherent that when an exception is detected, it will be detected at some point during the execution of a microinstruction. Therefore, it is the examiner's position that claim 16(a) is the same as claim 1(c) on page 10, lines 8-9, and claim 12(c) on page 11, lines 27-28. Therefore, claim 16(a) is rejected for the same reasons set forth in the above rejections of claim 1(c) on page 10, lines 8-9, and claim 12(c) on page 11, lines 27-28.

b) if an exception occurs, causing the exception to cancel all of the at least two of a plurality of microinstructions. From the rejection of claim 1(d) above, a person of ordinary skill in the art would have recognized that if one microinstruction causes an exception, and since all of the microinstructions represent a single macroinstruction, then all of the microinstructions should be cancelled so that an "undo" mechanism is unnecessary. This would result in saved processing time. Therefore, in order to achieve a more efficient system, one of ordinary skill in the art at the time of the invention would have realized that the exception should cause the cancellation of each microinstruction.

26. Referring to claim 17, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 12. It is noted by the examiner that claim 17 is the same as claim 6. Therefore, the rejection of claim 17 is based on the same reasons as mentioned above in the rejection of claim 6.

27. Claim 9 is rejected under 35 U.S.C. 103(a) as being unpatentable over Abdallah1 in view of Abdallah2 as applied to claims 1-8 above, and further in view of Phillips et al., U.S. Patent No. 6,038,652 (herein referred to as Phillips).

28. Referring to claim 9, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 1.

a) Recall that Abdallah2 has taught canceling the execution of the microinstructions if an exception occurs. See column 10, lines 21-24. In addition, it is well known in the art that if an unmasked exception occurs, a microcode handler must be invoked in order to correct the situation. This handler could be part of an operating system or implemented by a programmer. Therefore, in order to handle an exception that is not handled automatically by the processor, one of ordinary skill in the art at the time of the invention would have realized that a microcode handler must be implemented.

b) Abdallah1 in view of Abdallah2 has not specifically taught updating at least one exception flag (when an unmasked exception occurs) by independently generating a logical OR of exceptions for a plurality of functional units. However, Phillips has taught the concept of simultaneously checking SIMD elements for exceptions and combining each individual exception into an overall exception. See FIG.2. Furthermore, the combining element (230) in FIG.2 can be implemented as an OR gate that generates a flag (240) used to specify whether or not an exception has occurred. See column 3, lines 60-63. A person of ordinary skill in the art would have recognized that the concept of Phillips should be applied to the system taught by Abdallah1 in view of Abdallah2 in order to check for exceptions during the parallel execution of microinstructions. The exception flag alerts the processor that at least one exception has occurred and since the flag was produced according to a parallel execution as opposed to a serial execution, the processor will be able to cancel the instructions sooner if necessary. Therefore, it would have been obvious to one of ordinary skill in the art at the time of the invention to update at least one exception flag based on the exception check of each microinstruction.

29. Claim 18 is rejected under 35 U.S.C. 103(a) as being unpatentable over Abdallah1 in view of Abdallah2 as applied to claims 1-8 and 10-17 above, and further in view of Makineni et al., U.S. Patent No. 6,321,327 (herein referred to as Makineni).

30. Referring to claim 18, Abdallah1 in view of Abdallah2 has taught an invention as described in claim 17. Abdallah1 in view of Abdallah2 has not explicitly taught the use of two 82-bit FP registers used to hold two 32-bit single-precision floating-point values. However, Makineni has taught the use of 82-bit registers to hold two 32-bit single-precision floating-point numbers. See FIG.2B and FIG.3. A person of ordinary skill in the art would have recognized that variable register sizes could be implemented in Abdallah1's system. Recall that Abdallah1's system performs two 32-bit operations in parallel (See FIG.3). Therefore, the register size can be changed as long as multiple 32-bit values can be supplied to the adders. In addition, it would be beneficial to implement smaller registers in order to decrease the amount of overall hardware and wires. Therefore, in order to minimize the amount of hardware, it would have been obvious to one of ordinary skill in the art to implement two 82-bit floating-point registers (taught by Makineni) to emulate four 32-bit single-precision, floating-point values in an SSE register.

Conclusion

31. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure. Applicant is reminded that in amending in response to a rejection of claims, the patentable novelty must be clearly shown in view of the state of the art disclosed by the references cited and the objections made. Applicant must also show how the amendments avoid such references and objections. See 37 CFR § 1.111(c).

Rodgers et al., U.S. Patent No. 6,357,016 B1 has taught a floating-point execution engine that executes SSE instructions.

Thakkur, S. and Huff, T., in “Internet Streaming SIMD Extensions,” published in Computer, Vol.32, Iss.12, 1999, pages 26-34 has taught a SIMD architecture with packed, floating-point registers that handles exceptions.

32. Any inquiry concerning this communication or earlier communications from the examiner should be directed to David J. Huisman whose telephone number is (703) 305-7811. The examiner can normally be reached on Monday-Friday (8:00-4:30).

If attempts to reach the examiner by telephone are unsuccessful, the examiner’s supervisor, Eddie Chan can be reached on (703) 305-9712. The fax phone numbers for the organization where this application or proceeding is assigned are (703) 746-7239 for regular communications and (703) 746-7238 for After Final communications.

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the receptionist whose telephone number is (703) 305-3900.

DJH
David J. Huisman
August 20, 2002

Eddie Chan

EDDIE CHAN
SUPERVISORY PATENT EXAMINER
TECHNOLOGY CENTER 2100